Computational Culture
a journal of software studies

Creative Malfunction: Finding fault with Rowhammer

ARTICLE INFORMATION

- **Author(s):** Matt Spencer
- **Affiliation(s):** Centre for Interdisiplinary Methodologies, University of Warwick

- **Publication Date:** July 2021
- **Issue:** 8
- **Citation:** Matt Spencer. "Creative Malfunction: Finding fault with Rowhammer." *Computational Culture* 8 (July 2021). http://computationalculture.net/creative-malfunction-finding-fault-with-rowhammer/.

ABSTRACT

This essay offers a close reading of a notable computer security vulnerability: the Rowhammer bug in Dynamic Random Access Memory. The story of Rowhammer provides a vivid demonstration of how previously unanticipated possibilities of malfunction emerge as vulnerabilities and, in demanding repair, exert strong pressures on the future development of technologies. Vulnerabilities like Rowhammer, I show, do not appear fully formed; Rowhammer emerged over several years, as successive studies teased out the nature of the problem and invented new methods that put it to use in practical exploits, demonstrating how it can be used to compromise the security of affected systems. These studies cast new light on a variety of existing components, rendered faulty in relation to their failures to contain the error, or in their usefulness for the crafting of an exploit. There is no simple fix for a problem like Rowhammer. Any resolution depends upon a characterisation of the fault, which as we will see can still be subject to further revision years later. I close with an examination of some of the theoretical implications. The study of computer vulnerability, I argue, gives us insights into the historicity of technology and the permanent conditions of change and revision that characterise contemporary computing. Vulnerability research can be understood as a process of real time exploration of computational systems' 'adjacent possible,' creating new ways in which things can be at fault. Drawing on critical studies of repair, I argue that the interventions that arise from these explorations should be understood, not in terms of restoring a system to a prior good state, but as a creative, future-making force.

Introduction

Breaches, hacks and outages are in the news on a daily basis. Yet dependencies on digital technologies ever deepen. The distribution of renewable energy calls for smart grids, autonomous vehicles for machine vision, 'just in time' industrial production for software-driven orchestration. Exposure to computational vulnerabilities has become a quotidian feature of contemporary existence, but the nature of vulnerability itself is far from clear.

The relations between technology and failure have been a prominent issue for social theory for a number of decades, from normal accidents to the risk society[1], with attention renewed in recent years by a wave of critical studies of repair and maintenance[2]. But while a great deal of attention has been applied to accidents, disasters and breakages, to hackers and cultures of hacking[3], and to the nature of cyber attack[4], relatively little has been paid to the nature of vulnerabilities in computing technologies as such[5].

There is a crucial difference here: hacks, breaches and outages are events in which a system has *actually* behaved in an improper manner, while a vulnerability can be understood as a *possibility* of errant behaviour[6]. Even a computer vulnerability that has not in fact been exploited by malicious agents can exert tremendous pressures on the technology's later course of development, especially where it breaches fundamental logics in terms of which systems are considered to be secure. In this paper, I examine one recent and fundamental vulnerability, the Rowhammer problem with DRAM (Dynamic Random Access Memory), which presented alarming possibilities for a malicious program to manipulate data it should not have access to.

Rowhammer is one example among many. And its story remains unfinished. But there is enough here to gesture towards a general picture, one of digital technologies traversed by many concurrent trajectories of 'finding fault', processes that cast established structures and techniques in new lights, revealing previously unanticipated possibilities for (mis)use. In raising these issues of 'what is at fault', practitioners generate justifications for intervention and modification that inform the course of technical development. The challenge of thinking through vulnerability, then, is one of understanding how fault and failure can become, through these practices a creative, future-making force. Practitioners' research into computer vulnerabilities, then, is not just a source of external commentary on problems with technologies, but is, I will argue, a driving force of technological historicity, a process that articulates how things

ought to function against the potential invention of novel forms of malfunction, enfolding creative future-making with fault and failure.

I devote the initial four sections in what follows to laying out the story of Rowhammer's discovery and its immediate implications, the subsequent revelations about how it might be exploited, the kinds of repair that emerged, and their own relationships with the fault. I then offer some higher-level reflections about the recent 'turn to hardware' in vulnerability research, in which Rowhammer has been pivotal. I close the essay with a theoretical elaboration of the key themes this empirical object raises: the nature of failure, historicity, possibility and function.

Discovering Malfunction

Vulnerabilities are not only described by vulnerability research; they are made, they are *enacted* , in it. Something is identified as being at fault, in permitting improper behaviour, but its nature—what it is that is at fault, how it matters and what needs to be done as a remedy—is unfolded over sometimes considerable durations of time. The vulnerability undergoes a process of crystallisation, from its origins as a phenomenon capable of being pointed to and mustering attention, eventually emerging as a specified problem with existing or future systems, and exerting pressures on the formulation of future iterations. Malfunction and repair stand in a relation of mutual normative inflection: 'what is wrong' has profound implications for what would count as an 'appropriate' remedy[7].

The security analysts who populate the field of vulnerability research inhabit a position in relation to digital technology inverse, in certain senses, to that of system designers. The latter are concerned with specifying what is the correct behaviour for novel systems, a specification which serves both as template for building them and as measure against which they are judged. Security analysts, in contrast, are concerned with inventing methods for teasing out novel functionality that ought not to be possible, from systems that already exist. Their 'epistemic things' are previously unanticipated functional possibilities[8].

These vulnerability researchers, working in both commercial and academic settings, report their findings publicly through conferences, in blogs and academic papers, discursive exchanges that are akin to other fields of research in science and technology, albeit with two distinctive features. The temporalities of 'responsible disclosure' delay publication in certain circumstances, allowing

manufacturers a window of opportunity to pre-emptively fix a problem yet to be publicly announced. The field is also distinctive for the fact that these researchers are working in an antagonistic interaction with others, figures whose work their strongly resembles, but who pursue malfunction from within, for instance, nation state intelligence services, criminal organisations, or independently, extracting value from their finding vulnerabilities insofar as they then are able to keep them hidden from public knowledge.

Some vulnerabilities in digital systems are discovered across this divide, through the analysis of malware or cyber-attacks, known as 'zero days' if the vulnerability in question was unknown until the event of being encountered in use 'in the wild'. Others are discovered through exploratory research. Rowhammer is of this latter kind. It afflicts the Dynamic Random Access Memory (DRAM) chips that provide the working memory for devices from PCs and servers to smartphones. A group of vulnerability researchers from Carnegie Mellon University, investigating quirks of memory chip design, found that they were able to produce a rogue effect that contravenes the DRAM specification. This 'Rowhammer' problem was initially reported as affecting DDR3 (double data rate type three) chips which were in production since around 2007, but later studies noted that the vulnerability continued to be an issue in some DDR4 chips (the most recent standard) as well[9].

Brought to the security research community in 2014, these findings had huge potential significance[10]. However, at the point of its announcement, Rowhammer was not an *attack* per se, but rather a rogue possibility, a loose end whose implications were only partially understood. While the effect in question could be demonstrated, it was as yet unclear what it could be used for and in what circumstances. These questions drew the attention of a number of research groups, and prompted them to set out on endeavours to invent and examine ways that this particular effect could be put to use as a component part of exploits, looking in detail at the various contexts of contemporary computing, such as virtualisation technologies, different platforms, operating systems, processors, and so on[11].

The earliest references to Rowhammer, however, can be found in electronic engineering patent applications filed in mid 2012[12]. Engineers designing newer variants of DRAM chips were developing techniques to mitigate the effects of pushing chip architecture to ever smaller spatial scales. Memory in DRAM is composed of arrays of capacitors, accessed on a per row basis: the whole row is activated for any 'read' or 'write' operation to an individual capacitor it contains.

The state of charge of these individual capacitors ('high' or 'low') forms the material substrate that represents the 'stored' ('1' or '0') bits of information. It is the disturbance of this state that is at issue in Rowhammer, something that becomes more likely when the capacitors are smaller and more closely packed. A 'row' being 'hammered' describes the situation where a row of capacitors is accessed repeatedly in a brief interval of time. In certain cases, such patterns of repeated activation can induce an error in a nearby memory location in a different row. 'A specific row of a memory device can be the target of repeated accesses. When the row is accessed repeatedly within a time threshold (also referred to as 'hammered' or a 'row hammer event'), a physically adjacent row or rows (a 'victim' row) may experience data corruption'[13].

The Carnegie Mellon researchers' discovery was that these effects could in fact be induced on commodity hardware, chips being sold in the market and, at that point, in use within countless devices. Using a specially crafted program, which systematically 'hammers' memory locations, they were able to find vulnerable locations on commodity chips made by all major manufacturers.

The Carnegie Mellon group made their findings public in a paper entitled 'Flipping Bits in Memory Without Accessing Them'[14]. The fundamental idea, expressed in that title, is profound: that it is possible to modify contents of memory *without accessing* the bits in question. The dominant paradigm of computer security architecture depends upon memory isolation to ensure that processes of different levels of trust can safely run concurrently on the same machine. A process is allocated 'its' memory and should not be able to read or write to anything else. The state diagram in the DDR3 specification shows that *only* through being accessed, i.e. via the activation of the row, should it be possible for data in memory to be modified or read[15]. It is the control of this access that is supposed to guarantee control of data. But that is entirely circumvented if a rogue form of writing is possible, writing without access, breaking the fundamental promises of the design.

Containing Malfunction

'If a cell is disturbed beyond its noise margin,' Kim et al. wrote, 'it malfunctions and experiences a disturbance error'[16]. However, disturbance errors, they note, are as old as DRAM itself. The issue was that these errors were—wrongly—thought to be comprehensively mitigated by existing techniques. For mitigation, Kim et al write, 'DRAM manufacturers have been employing a two-pronged approach: (i) improving inter-cell isolation through circuit-level techniques and

(ii) screening for disturbance errors during post-production testing. We demonstrate that their efforts to contain disturbance errors have not always been successful, and that erroneous DRAM chips have been slipping into the field'[17].

It was always more than one thing that was faulty. The malfunction may be named as a disturbance error, but it is a problem insofar as this disturbance is accompanied by a failure to successfully *contain* these errors within the noise margin of the chip, and indeed a failure of screening in the factory to pick up defective goods. Kim et al. give us a helpful reminder that a kind of 'double vision' is needed, that we must not fixate on microscopic goings on in silicon chips alone, but also consider factory logistics and quality control. Just as the DDR3 specification is used in developing chips and configuring fabrication machinery, it is also used to devise testing apparatuses for quality control. Is this the problem? Of faulty goods exiting the factory through the wrong door, as commodities rather than as waste by-product? The authors reveal themselves to be masters of dramatic understatement, for 'all modules [in their sample] manufactured in the past two years (2012 and 2013) were vulnerable'[18]. Erroneous DRAM chips weren't 'slipping' into the field, they were pouring!

In addition to the mitigations represented by the physical design of chips and screening post-manufacture, we can add a third prong, which Kim et al do not refer to as a mitigation because it is the foundation of Random Access Memory in the first place. States of 'high' and 'low' charge are defined relative to thresholds of measurement that render variances of charge (within these thresholds) irrelevant to the higher-level processes that 'use' them as digital data. This is important because in DRAM charge does not stay put; it continuously 'leaks'. It is the fate of 1s to become 0s, or if we consider memory as a signal sent through time, a falling away in the discernibility of information from noise. DRAM is thus memory (as opposed to a kind of momentarily persistent electrical afterimage) insofar as it is 'refreshed' with a regular and rapid cadence, differences restored faster than they can dissipate.

Refreshing memory requires dedicated hardware: a refresh counter sits on the chip alongside the banks of capacitors. But it is also a programmatic operation handled by the processor's memory controller. Alongside the core commands for the chip: 'access' and 'read' and 'write,' the chip provides the 'refresh' command. The DDR3 specifications, Kim et al. tell us, 'guarantee a retention time of at least 64 milliseconds, meaning that all cells within a rank [a rank is a group of rows] need to be refreshed at least once within that window'[19]. The

locus of the fault thus expands, for this refresh mechanism may also be considered at fault, too sluggish to save potential victims from the effects of hammering before they get disturbed.

It is from this kind of lockstep between disturbance and operations that disambiguate charge that the digitality of memory is produced. Noise, leakages, disturbances are not errors in themselves but become errors in relation to the mechanisms that are supposed to contain them, to render them irrelevant to the higher-level configurations that make use of the chip's signals as digital memory. The containment of error, which is at first glance a second-order mechanism, at second glance is recognised as a constitutive element in the implementation of digitality itself. As Blanchette observes, '[i]t is this ability to ceaselessly cleanup after its own noise that so powerfully enables computers to seemingly sever their dependency on physical processes that underlie processing, storage, and connectivity'[20].

Abstraction in computation is also a social technology, in the sense of separating the promises a component makes from the details of its implementation. A software programmer does not need to be considering the dissipation of charge in capacitors when they define their variables. Indeed the aggregation of levels and layers of abstraction that characterises computing is an effect of effective constraint on relevance. The operating system, processor, and memory chip are trusted to handle the details. But these very divisions of attention and labour in technical structure[21], can themselves be turned to illicit use: 'abstraction,' writes Schaefer, 'is the cover from which one hides an attack'[22]. The refresh cycle is supposed to maintain cells' state of charge, the chip geometry is supposed to keep such leakages of charge within tolerable thresholds, quality control is supposed to pick up defective chips. Software developers, or at least the vast majority of them, are not supposed to have to worry about this. The division of attention that enables them to think in terms of data rather than the state of charge of electronic components, is in turn revealed to be at fault, at least insofar as it is this abstraction that enables an adversary to 'hide' the possibility of attacks from them, in layers below.

This simple consideration of 'what is at fault' in the basic initial phenomenon of Rowhammer shows us there is no simple fault. We see instead a kind of 'refractive' normativity: the problem is not simple and singular, but refracts through the technical structure, various components appearing at fault in terms of their relations with others. In the next two sections we see how this complexity develops through progressive research into what might be *done with*

Rowhammer and then, what might be *done about* it, the implications for exploitation and repair. The timescales are different but the pattern is familiar: just as the refresh cycles maintain digital abstraction over millisecond durations, so too, over weeks, months, and years, do the projects of vulnerability research maintain function in the face of new possibilities for errant behaviour.

Exploiting Malfunction, Crafting Exploits

The simplest result that can be achieved with Rowhammer would be causing the computer in question to crash, for instance if the bit flip affected a critical area of memory. An induced malfunction like this may have effects that ripple through the further technical systems to disastrous effect: a plane's autopilot, a factory's control systems, the braking systems of a modern automobile.

These effects are what Luciano Floridi and collaborators refer to as 'negative' malfunctions: things failing to do what they are supposed to do[23]. At the heart of an effective exploit, however, is often what they call a 'positive' malfunction. Where a negative malfunction is a failure to perform, a positive malfunction produces an inappropriate additional effect. It is often the prerogative of a malicious hacker to ensure that the core functionality of the compromised system is maintained intact, so that additional, off-specification functionality can be extracted 'on the side,' using the error for instance to craft a whole new interface capable of serving up illicit functionality, such as taking control of the device through 'escalation of privilege' and indeed doing so without attracting attention.

Within days of the presentation of 'Flipping Bits', researchers at Google's security analysis group *Project Zero* were reproducing the Carnegie Mellon results on their own test machines, and later reported on the basis of these tests that Rowhammer could be indeed used to craft a practical exploit to achieve privilege escalation[24]. In one of their examples, they used the technique to escape the Native Client (NaCl) sandbox in Google's Chrome browser, with the implication that code from a website could potentially take control of the computer.

Repair work spins off directly from these exploratory efforts. In their NaCl escape, for instance, Seaborn and Dullien made use of the CLFLUSH command (an x86 instruction), which provided a simple method for meeting one of the central pragmatic challenges for crafting a Rowhammer exploit; to ensure that each strike of the 'hammer' hits the row in the memory chip, it is necessary to

circumvent the processor's intermediate levels of cache. For this reason, a November 2014 release of Chrome removed the ability to call CLFLUSH from within NaCl, later disclosed as CVE-2015-0565. As it unfolds, a vulnerability can thus cast new light upon existing components. Although CLFLUSH was doing what it was supposed to do, its usefulness in crafting a Rowhammer-based exploit implicated it, such that its availability in NaCl became a bug calling for a fix.

It is not just 'naïve' elements that prove useful for rogue purposes. Techniques explicitly devised as security mitigations in one respect may provide functionality that turns out to be helpful to an exploit in another. Aga et al. for instance, found that circumventing the cache was more easily done where chips used Intel's 'Cache Allocation Technology' (CAT). CAT is supposed to suppress the ability for co-tenanted virtual machines (for instance in cloud infrastructures) to interfere with one other's performance by manipulating their shared cache. But the constraint it places on a process's use of the cache made it easier to achieve the high rates of memory access necessary for hammering, a result the authors use to argue for 'subtractive' rather than additive responses to security issues: even security controls can prove useful in unforeseen ways, for new and illicit purposes[25].

A second pragmatic issue entailed in crafting a Rowhammer exploit is the need to manipulate where sensitive data is placed in memory, so that a *useful* bit can be flipped. Vulnerable chips may only have a small number of memory locations that are sensitive to the technique, so the challenge is controlling what data ends up in those positions, to be corrupted in the attack.

Rowhammer thus stimulated innovation in 'memory massaging' techniques. It was often techniques devised originally for performance optimisation that turned out to be useful here, enabling a hostile process to exert control on the placement of data in memory. For instance, the Amsterdam-based VUSec group's 'Flip Feng Shui' technique relies on memory deduplication, an optimisation technique commonly used in virtualised cloud environments[26]. Deduplication increases the overall amount of memory available by identifying duplicate segments of data (which may be 'owned' by entirely different processes) and freeing up spare memory by storing only a single copy. Whenever a process writes to 'its' data, a separate copy is created, preventing that 'write' operation from affecting other processes' data. Razavi and colleagues describe using Rowhammer to surreptitiously introduce changes into deduplicated memory. Because the memory was modified without a write

operation, they were able to sneak changes into memory owned by other processes. In their example, they report using this technique to alter cryptographic keys, thus weakening the resulting encryption[27].

A second example of 'memory massaging' comes from VUSec's 'Drammer' (deterministic Rowhammer) project, which targets Android smartphones. Here they made use of the memory allocator algorithm used in Android Linux, which is designed to allocate memory efficiently[28]. This 'buddy allocator' algorithm was developed in the 1960s at Bell Labs[29]. Originally named the 'fast storage bookkeeping method' it trades off speed against capacity and minimises fragmentation by allocating memory to processes in blocks of predetermined sizes, chosen so that the sizes nest neatly into each other (typically either powers of 2 or Fibonacci numbers). By carefully flooding and releasing specially chosen sized blocks of memory, the VUSec group devised a method to exert fine control over which regions of physical memory are 'free,' and thus predictably induce sensitive memory to land in a targeted location. In their exploit, they induce the operating system to place a 'page table page' in a vulnerable location: this is the record upon which memory isolation is based, the record of which processes have access to which 'pages' of memory. By corrupting this data, their Android app was able to gain control over the device, a complete subversion of the phone's security model.

These processes of devising exploits make the vulnerability concrete. Firstly, in demonstrating practical techniques that go beyond a proof of concept, they demonstrate what kinds of illicit activities the vulnerability might be useful for. Secondly, because exploits are technical systems in their own right, a broader set of normativities are perturbed. Repurposed for memory massaging, components like the buddy allocator gain a split normative personality: it means one thing to be working properly for Drammer as a means of memory massaging and, in being repurposed as such, another to be working improperly for the phone or Linux in enabling this activity. Even elements which have been stable for many decades may have their function cast in a new light as wider reconfigurations shift around them.

Repairing Function

Each contribution to vulnerability research has implications for what counts as an appropriate mechanism for mitigating Rowhammer. Repair's own normativities (what might constitute the right kind of fix) are closely aligned to the determination of the malfunction, and both are provisional and tentative.

For Rowhammer, a number of fixes have been suggested[30]. Some, such as increasing the rate at which rows are refreshed, have performance implications but can be implemented on existing hardware. Others concern how newer chips ought to be made in order to prevent this kind of problem occurring in the future. More peripheral fixes also spin off as a result of how the vulnerability reshuffles normativity for systems that end up implicated in some way. Removing the CLFLUSH command from NaCl, for instance, does not solve the bigger problem of Rowhammer, but makes implementations of exploits more difficult, and avoids NaCl being implicated as a source of assistance. But mitigation mechanisms are themselves functional artefacts, with their own adjacencies. How do you ensure that the very enactment of a 'fix' doesn't become fresh cover for an attack?

The primary response adopted by DRAM manufacturers in light of this work has been the introduction of 'Targeted Row Refresh' (TRR) to newer variants of their chips. TRR adds a mechanism to monitor patterns of access to memory. Where it detects something suspiciously Rowhammer-like, it will trigger a refresh operation for nearby rows, heading off the possibility of disturbance errors from being induced.

The chip makers, however, designed TRR to spot typical patterns of hammering. Manufacturers addressed the problem based on what it *usually* looks like: patterns either of single-sided or double-sided hammering, in which rows either on one or both sides of a victim row are hammered. In 2020, however, several prominent figures in Rowhammer research published a tool they named 'TRRespass', which identifies novel patterns of hammering which circumvent implementations of TRR[31]. TRRespass is a 'fuzzer,' a reference to a wider class of security analysis tool that fires randomly generated inputs at a component in a probabilistic search for previously unanticipated ways to induce errant behaviour. TRRespass searches through *atypical* patterns of hammering to identify patterns that cause bits to flip yet are invisible to TRR implementations. And out of 42 supposedly Rowhammer-proof modules they tested, they found effective patterns on 13[32]. This demonstrates the mutually enfolded openness of vulnerability and repair. Only in hindsight will a fix have held steady.

But perhaps the most striking example of the unsettled openness of malfunction is the recent announcement of 'RAMBleed' by an international group of collaborators based across the University of Michigan, TU Graz, and The University of Adelaide[33]. Their paper's subtitle riffs on Kim et al.'s original 'Flipping Bits': they named it 'Reading Bits in Memory Without Accessing Them'.

The fuss around Rowhammer was about writing, the ability to change bits in memory that didn't belong to you. With attention fixated on problems of illegitimate writing, the production of a novel form of reading goes back to basic questions about the very identity of Rowhammer and thus of what counts as a fix. The RAMBleed researchers had returned to the basic question of what could be done with flipping bits, considering the case where the memory locations concerned are within your own (i.e., the hostile process's) control. On first glance, this is a strange exercise. They already control these 'victim' rows, so they could just write new values to them using the conventional means. Why go to the trouble of flipping them with the extreme methods of Rowhammer, using thousands of times more operations to achieve the same end? Kwong et al. found that by statistically analysing how readily hammered bits flip, they could make inferences about the state of charge in nearby memory locations, which can include, crucially, cells outside of their control, memory allocated to other processes.

One of the prime candidates for preventing Rowhammer exploits from being effective is the use of Error Correcting Code (ECC). The appropriateness of this kind of approach as a repair is based on the identification of Rowhammer as a process of rogue writing: ECCs make Rowhammer irrelevant to higher level processes through increased fault tolerance, being able to correct corrupted data on the fly. But as with TRR, the idea of a solution can itself become problematic due to the assumptions it embodies about *the nature of the problem*, assumptions which may as yet become unsettled.

RAMBleed makes use of the observation that Rowhammer effects are stronger in some circumstances than others. A low charge '0' bit is more likely to flip if the cells in adjacent rows have high charge '1' values, and vice versa. It is slightly easier in other words to change the state of a victim cell to match those of its neighbours than to change it to be different. So, by controlling the values of some neighbours, and observing over many iterations how sensitive to flipping a given cell is, the value of a far side neighbour can be inferred. Combining this technique with memory massaging to manipulate sensitive data into that neighbouring location, the group were able to extract a 2048-bit private SSH key from memory, without having touched, by hammering or otherwise, any memory outside of their own allocated pages, and without having conducted any privilege escalation. Because all bit flips occur within memory 'owned' by the hostile process, there is nothing for an ECC to correct. The very identification of the fault with writing becomes the fulcrum for a new reading-based attack.

That even a well-known and widely studied bug like Rowhammer might be revealed, 6 years after its original discovery, to manifest a radically new form, is a powerful demonstration of the openness to interpretation of what is at fault, with it what might count as a repair, and how technical development ought to continue. Seen in hindsight, the 'fix' may seem to be whatever closed down the issue, but in prospect, the 'fix' is not a closure at all, but rather the continuation of these same processes: pursuing the malfunction into new configurations, new normative diffractions of what is at fault and how it might be repaired.

Material Anxieties

Rowhammer drew attention to hardware exploits, in particular because it vividly demonstrated the potential of using programmatic means to induce them. Before Rowhammer, exploiting hardware vulnerabilities had typically required access to the machine, for example, being able to measure the power consumption, monitor the sound, or observe the blinking of an LED. Requiring physical access, those attacks don't scale in the way that software-based attacks can: you might deploy malicious code to millions of machines in the space of time only a handful could be physically accessed (and with considerably less risk to one's own person). The Google researchers Seaborn and Dullien pointed to a 2003 study by Govindavajihala and Appel as a precursor to Rowhammer. This study involved a demonstration that a program could be specially crafted so as to be extremely sensitive to bit flips in memory and capable of using them to escape from a Java Virtual Machine[34]. To achieve a bit flip, however, they either needed physical access (the researchers describe using a clip-on lamp with a spotlight bulb to heat the hardware sufficiently to induce data corruption) or else abundant patience, awaiting the intervention of a cosmic ray. We can thus read a double meaning to 'flipping bits in memory without accessing them' (the title of the original Rowhammer security paper): the circumvention of the computer's access control in the sense of access to memory pages; and secondly the circumvention of physical access control implied by the software-induced nature of the technique.

In channelling attention to possibilities of rogue function, particular projects influence the wider landscape of vulnerability research. Mutlu and Kim, two of the authors of the original 'flipping bits' paper, suggest that the surge of interest that followed Rowhammer may have been a factor in the subsequent discovery of the Meltdown and SPECTRE hardware vulnerabilities, reported in early 2018[35]. SPECTRE and Meltdown received a huge amount of attention in the press because these techniques had in theory been possible for decades

(though whether anybody knew this until 2018 is another question). These exploits abuse performance optimisation methods built into CPUs in order to extract sensitive data to which access should not be possible. Like Rowhammer they undermine memory isolation, and like Rowhammer they do so by attacking basic chip-level processes with software.

Given that the closure of repair is always to some extent deferred, anxieties remain, and they can crystallise around various aspects of technology. Soon after the SPECTRE and Meltdown announcements, a controversial Bloomberg feature entitled 'The Big Hack' made the extraordinary claim that additional chips had secretly been added onto widely used Intel circuit boards during manufacturing[36]. While that report remains disputed (and widely denied by the companies involved), the feasibility—and challenge—of detecting such additions has been vividly demonstrated[37]. What kinds of quality control can mitigate the possibility of rogue functionality being literally soldered on? Freshly posed questions about *what is there* in computer hardware, and what it can (be made to) do form the context for recent controversy around the use of equipment manufactured by the Chinese Huawei corporation for building 5G telecommunications infrastructure in Western countries. While geopolitics of course have a great deal to do with this, we may attribute some role to rising anxieties about the material implementation of computing, and the openness of what is possible with computational hardware.

Malfunction, normativity and the possible

While Rowhammer concerns the micro temporalities of hammering and dissipation of charge, it is constituted through a drawn-out process of unfolding uncertainties. This unfolding presents a striking methodological opportunity for empirical engagement with theoretical questions around the nature of malfunction, of normativity and the possible. In this closing section I address these themes more directly.

The invention of a vulnerability is hard to disentangle from the invention of its repair: the two processes are mutually implicated, and both entail considerable uncertainty about whether and how they will stand the tests of time. Vulnerability research thus does not only aim at describing vulnerabilities, but also at 'looping' around and changing its object such that such rogue possibilities can be closed off. In this sense we might see an analogy between vulnerability research itself and processes such as the refresh of charge on a

DRAM chip: while operating over very different timeframes, both processes do work of 'containing' noise within tolerable margins.

As stressed above, vulnerability research does not only respond to things that have actually broken; it also involves the invention of new methods and new possibilities. It resembles what Bachelard termed a 'Philosophy of No'[38]: It works upon and against assumptions that have conditioned technical development in the past, and that are now materialised in it, teasing out previously unanticipated effects. In Bachelard's description, science likewise is to be seen as intrinsically processual and articulated against itself. 'The intellectual life of science' he wrote, 'depends dialectically on [the] differential of knowledge at the frontier of the unknown'[39]. Bachelard's philosophy of knowledge was pivotal in establishing the tradition of historical epistemology in 20th century thought. The implications of vulnerability research are in contrast better understood in terms of historical *ontology*, a differential of *intervention* at the frontier of the unknown. The term is, of course, already prefigured in my use above of the term 'looping', for 'historical ontology' was the name given by Ian Hacking for his interest in 'looping' effects of human kinds[40]. For Hacking, kinds of person, such as those with 'multiple personality disorder' are both described and made in psychiatric science. For all that vulnerability research is worlds away from psychiatry, it too involves processes in which description is inextricable from self-making. The explication of Rowhammer describes and creates the vulnerability in DRAM, establishing new terrain for its future.

In many ways, what we observe here is another angle on the iterativity of digital technologies, which has been widely remarked upon over the past two decades[41]. Further, the capacity for description and action to be conjoined in technical apparatuses was at the heart of discussions around the performativity of code that characterised early software studies[42]. But as Bogost has pointed out, there is a danger that the image of 'automation' implicit in software 'doing what it says' draws critical attention away from the conditions in which software as text can have this effect[43]. Vulnerabilities such as Rowhammer point to just such conditions: it is only insofar as the hardware is functioning correctly that software's performativity can hold steady.

If the functioning of computational systems of all kinds, then, rests upon processes of finding and repairing vulnerabilities, repair-work would be at the heart of computing in a way that the formalist interpretations tend to gloss over. Critical studies of repair are particularly useful here. Steven Jackson, for instance, has worked hard to displace the entrenched notion of repair as a

matter of restoration to a prior 'good' state, in favour of a more creative, future-making image. In one of Jackson's poetic flourishes, repair 'fills in the moment of hope and fear in which bridges from old worlds to new worlds are built, and the continuity of order, value, and meaning gets woven, one tenuous thread at a time'[44]. Weaving continuities is what vulnerability research is all about, finding the threads of ways things may not do what they are supposed to do, and tracing them through to interventions that follow, weaving the continuity of right ways to function.

But while his characterisation of repair is so helpful, the notion of failure that Jackson works with needs some elaboration, particularly where metaphors of mess and entropy drive intuitions of a natural, or default tendency of things to break down. 'Attending to breakdown points us toward the active and ontologically productive nature of ruin, and the irreducible presence of embedded materialities with rhythms and propensities all their own, which can only ever be sometimes, and for a while, slowed, arrested, and aligned'[45]. The risk here, emphasising 'propensities all their own', is of treating malfunction as a given, a self-evident occurrence happening of its own accord. It is certainly true that some faults arise from this kind of event. But others are the products of active investigation, teased out through painstaking and creative work.

A related challenge for a theory of vulnerability and malfunction is the tendency in social studies of infrastructures to examine breakdown in terms of the consequences that follow failure, rather than the circumstances of its production. Breakdown is associated with the becoming visible of the ordinarily invisible. In Star and Ruhleder's classic account 'The normally invisible quality of working infrastructure becomes visible when it breaks; the server is down, the bridge washes out, there is a power blackout. Even where there are back-up mechanisms or procedures, their existence further highlights the now-visible infrastructure'[46] the point where the system "undoes itself" is a malfunction, something that needs to be fixed. From the perspective of deconstruction, in turn, it is a point of revelation, one in which the conceptual system underlying the software is clarified' Frabetti, F. 2014. *Software Theory: A Cultural and Philosophical Study*. London: Rowman and Littlefield International, 70.]. While vulnerability research might be thought of as a back-up mechanism of the most flexible kind, entirely bespoke to the problem in hand, this does not do justice to the fact that those processes drew out the novel possibility in the first place, rather than (just) responding to it.

To engage with the drawing out of novel possibility, this 'frontier of the unknown,' to recall Bachelard's words, is itself a challenge. Rowhammer can easily be interpreted as the discovery of a design oversight, and the constant flow of new vulnerabilities as our encounter with the imperfection of human capacities to master the technologies we create. A useful argument is offered by Stuart Kauffman, who has suggested that the realm of the possible in which novel function may be found (by evolution), is in many cases 'non-prestatable'[47]. There is no way to map in advance a 'non-prestatable" space, no shortcut for evolving systems (understood very broadly) but to explore their 'adjacent possible' in real time. As Kauffman has argued with collaborators in a recent paper explicitly linking this claim to the genesis of novel technical function, what is missing is the problem of the emergence of novel fitness in the first place.

'There are adjacent possibilities and niches for each trait, function, or capability of an organism and new organisms may be —in the terminology of Longo et al. (2012)—'enabled.' It is not possible to map all of these possible adjacencies, just as all the uses of a screwdriver are not algorithmically listable, nor are all the opportunities that arise listable or prestatable. Moreover, all that must occur in evolution is that some molecular screwdriver in an evolving organism 'finds a use' that enhances the fitness of the organism and that there be heritable variation for that use. Natural selection might then positively select for this newly adapted use. This is the arrival of the fitter, missing in selection-oriented approaches.' [48]

The process of devising exploits entails just such a search for adjacencies, assembling methods out of functionalities that had been put in place for other reasons, for new purposes like circumventing the cache and for memory massaging. It is at the interstices of a complex assemblage of functional possibilities that new rogue function emerges, that catches attention, and persists in a trajectory of inquiry, drawing out new methods, possibilities and consequences, in turn shifting that whole assemblage into a new configuration. It wouldn't be necessary to abandon the intuition that, in many cases, designers should have foreseen the vulnerabilities that eventually emerged, to conceptualise the wider space of possibility in relation to which they are working, as non-prestatable. Doing so allows for a radically different relationship with temporality: exploration of the possible is a real time endeavour, at the heart of the historicity of function.

In the background here, and throughout this discussion, is a normative approach to function I adopt in order to stay close to normal language: Rowhammer describes a phenomenon that is *not right* about DRAM, a effect that is *incorrect*, that *should not* be possible. Research around Rowhammer is initiated by the demonstration that it is possible to create an effect that ought not be possible. But importantly, as we have seen, many engineered systems operate through systems of fault tolerance rather than purely through the removal of fault. The tolerance of error implies a relational structure in which some parts are supposed to render certain kinds of deviation irrelevant to others. It is this relational structure that the study of exploits and repair in relation to Rowhammer traverses. Components are revisited that have long endured, such as the Buddy Allocator, and so are recent fixes such as Targeted Row Refresh. Vulnerability research thus produces new normative refractions through the histories established within infrastructure, both in terms of what counts as problematic and in terms of what counts as a suitable repair.

The concept of function has, of course, a fraught and contested history. Function*alism* describes an explanatory principle of great importance in the 20th century social sciences: the explanation of social phenomena according to their causal role in reproducing society. It also describes one of the most influential aesthetics in 20th century design and architecture. Philosophers aiming to produce a theoretical analysis of the concept have commented on the multiple ways it can be grounded; for some these are theoretical alternatives, for others positions to be integrated. An analysis of the concept is notoriously complex, requiring accounting for intentions of designers and users, for causal structures and conditions, and for histories of selection and reproduction[49].

The normative approach adopted here does not aim to crystallise the essence of function as a theoretical resource, but rather to attend to the practices of vulnerability research. If ascertaining what things ought to do is an endeavour of 'artifact hermeneutics'[50], my interest here has not been in applying artifact hermeneutics as an analytical strategy, but rather to examine the artifact hermeneutics performed in vulnerability research and at the heart of processes of change and stasis of computational systems. The elicitation of what is possible, the interpretation of what is wrong and the formulation of what can be done about it, are processes internal to computational infrastructures. Function would then appear neither as an ideal state we've lost nor as one at which we failed to arrive, but as a moving image of maintenance, self-articulated processes that invent new ways to fail, to foster repair and find fault.

Author Biography

Matt Spencer is an Associate Professor at the University of Warwick's Centre for Interdisciplinary Methodologies. His current research focusses on the nature of trust in the technical practices of cyber security. He is a UK Research and Innovation Future Leaders Fellow.

Contact Details

m.spencer.1(AT)warwick.ac.uk

Acknowledgements

I would like to express my gratitude to Celia Lury, Naomi Waltham-Smith, Maria Puig de la Bellacasa, Andrew Goffey and two anonymous reviewers, whose insightful comments made a big difference to the process of writing this essay.

This research received support from a UK Research and Innovation Future Leaders Fellowship.

Bibliography

Aga, M. T., Z. B. Aweke, and T. Austin. 2017. "When Good Protections Go Bad: Exploiting anti-DoS Measures to Accelerate Rowhammer Attacks." *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)* , 8-13

Amoore, L. 2013. *The Politics of Possibility: Risk and Security Beyond Probability*, Durham: Duke University Press

Bachelard, Gaston. 1968. *The Philosophy of No: A Philosophy of the New Scientific Mind*. Viking Press

Bachelard, G. 1984. *The New Scientific Spirit*. Trans. A. Goldhammer. Boston: Beacon Press

Bains, K., J. Halbert, C. Mozak, T. Schoenborn, and Z. Greenfield. "Row Hammer Refresh Command." U.S. Patent 9,117,544, issued August 25, 2015.

Beck, U. 1992. *Risk Society: Towards a New Modernity*. London: SAGE

Blanchette, J.-F. 2011. "A material history of bits." *Journal of the American Society for Information Science and Technology* 62, 6:1042-1057

Bogost, I. 2015. "The Cathedral of Computation", *The Atlantic.* Available at: https://www.theatlantic.com/technology/archive/2015/01/the-cathedral-of-computation/384300/

Coleman, G. 2012. *Coding Freedom: The Ethics and Aesthetics of Hacking*. Princeton: Princeton University Press

Coleman, G., and C. Kelty. 2017. "Hacks, Leaks, and Breaches." *LIMN* 8

Chun, W. H. K. 2011. *Programmed Visions: Software and Memory*, Cambridge: MIT Press.

Chun, W. H. K. 2016. *Updating to Remain the Same: Habitual New Media*, Cambridge: MIT Press

Dennett, D. C. 1990. "The Interpretation of Texts, People and other Artifacts." *Philosophy and Phenomenological Research*, 50: 177-194

Dwyer, A. C. 2019. *Malware Ecologies: A Politics of Cybersecurity*. Doctoral dissertation: University of Oxford.

Elkins M. 2019. "Nation-State Supply Chain Attacks for Dummies and You Too." *CS3STHLM* 21-22 Oct. 2019.

Felin, T., S. Kauffman, R. Koppl, and G. Longo. 2014. "Economic Opportunity and Evolution: Beyond Landscapes and Bounded Rationality." *Strategic Entrepreneurship Journal* 8, 4: 269-282

Floridi, L., N. Fresco, and G. Primiero. 2015. "On Malfunctioning Software." *Synthese* 192, 4: 1199-1220.

Frabetti, F. 2014. *Software Theory: A Cultural and Philosophical Study*. London: Rowman and Littlefield International

Frigo, P., E. Vannacci, H. Hassan, V. van der Veen, O. Mutlu, C. Giuffrida, H. Bos, and K. Razavi. 2020. "TRRespass: Exploiting the Many Sides of Target Row Refresh." *IEEE Symposium on Security and Privacy* 747-762

Galloway, A. R. 2006 "Language Wants to be Overlooked: On Software and Ideology." *Journal of Visual Culture*, 5, 3: 315-331

Govindavajhala, S. and A. W. Appel. 2003. "Using Memory Errors to Attack a Virtual Machine." *IEEE Symposium on Security and Privacy*, 154-165

Graham, S., and N. Thrift. 2007 "Out of Order: Understanding Repair and Maintenance." *Theory, Culture & Society* 24, 3: 1-25

Greenfield, Z., K. S. Bains, T. Z. Schoenborn, C. P. Mozak, and J. B. Halbert. "Row hammer Condition Monitoring." U.S. Patent 8,938,573, issued January 20, 2015.

Hacking, I. 2004. *Historical Ontology*, Cambridge: Harvard University Press

Hayles, N. K. 2016. "Cognitive Assemblages: Technical Agency and Human Interactions." *Critical Inquiry* 43, 1: 32-55.

Hommels, A., Mesman, J. and Bijker, W.E. 2014. *Vulnerability in Technological Cultures: New Directions in Research and Governance.* Cambridge: MIT Press

Houkes, W., & Vermaas, P. E. 2010. *Technical Functions: On the Use and Design of Artefacts* London: Springer

Jackson, S. J. 2014. "Rethinking Repair." T. Gillespie, P. J. Boczkowski, K. Foot (eds.) *Media Technologies: Essays on Communication, Materiality, and Society* 221-239

Jackson, S. J. 2017. "Speed, Time, Infrastructure." J. Wajcmann, N. Dodd (eds.) *The Sociology of Speed: Digital, Organizational, and Social Temporalities* 169-186

Joque, J. 2018. *Deconstruction Machines: Writing in the age of cyberwar*. Minneapolis: University of Minnesota Press

JESD79-3F DDR3 SDRAM Standard (2012)

Kauffman, S. 2000. *Investigations*. Oxford: Oxford University Press

Kelty, C. 2008 *Two Bits: The Cultural Significance of Free Software*. Durham: Duke University Press

Kim, Y., R. Daly, J. Kim, C. Fallin, J. Hye Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. 2014. "Flipping Bits In Memory Without Accessing Them: An Experimental Study Of DRAM Disturbance Errors." *ACM SIGARCH Computer Architecture News* 42, 3: 361-372.

Kittler, F. 1995. "There Is No Software." *CTheory* 10-18.

Knowlton, K. C. 1965. "A Fast Storage Allocator." *Communications of the ACM* 8, 10: 623-624.

Kocher, P., J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg et al. 2019. "Spectre Attacks: Exploiting Speculative Execution." *IEEE Symposium on Security and Privacy* 40

Kwong, A., D. Genkin, D. Gruss, and Y. Yarom. 2020. "Rambleed: Reading bits in memory without accessing them." *IEEE Symposium on Security and Privacy* 41

Lipp, M., M. Schwarz, D. Gruss, T. Prescher, W. Haas, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg. 2018. "Meltdown: Reading Kernel Memory from User Space." *27th USENIX Security Symposium*. 18: 973-990.

Mackenzie, D., and G. Pottinger. 1997. "Mathematics, Technology, and Trust: Formal Verification, Computer Security, and the US Military." *IEEE Annals of the History of Computing* 19, 3: 41-59.

Mutlu, O. and J. S. Kim. 2019. "RowHammer: A Retrospective." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*

Neff, G., and D. Stark. 2004. "Permanently Beta." In Howard P. N. and Jones S. (eds.) *Society Online: The Internet in Context* London: SAGE. 173-188.

Perrow, C. 1984. *Normal Accidents: Living With High Risk Technologies*. Princeton: Princeton University Press

Razavi, K., B. Gras, E. Bosman, B. Preneel, C. Giuffrida, and H. Bos. 2016. "Flip Feng Shui: Hammering a Needle in the Software Stack." *USENIX Security Symposium* 25: 1-18

Rheinberger, H.-J. 1997. *Toward A History of Epistemic Things: Synthesizing Proteins in The Test Tube*. Stanford: Stanford University Press

Robertson, J., and M. Riley. 2018. "The Big Hack: How China Used a Tiny Chip to Infiltrate US Companies." *Bloomberg Businessweek* 4

Rouse, J. 2016 "Normativity". In J. Kiverstein (Ed.) *The Routledge Handbook of Philosophy of the Social Mind*. London: Taylor & Francis. 545-561

Seaborn, M., and T. Dullien. 2015. "Exploiting the DRAM Rowhammer Bug to Gain Kernel Privileges." *Black Hat* 15

Schaefer, R. 2009. "The Epistemology of Computer Security." *ACM SIGSOFT Software Engineering Notes* 34, 6: 8-10.

Star, S. L., and K. Ruhleder. 1996. "Steps Toward an Ecology of Infrastructure: Design and Access for Large Information Spaces." *Information Systems Research* 7, 1: 111-134.

Stiegler, B. 2012. "Relational Ecology and the Digital Pharmakon." *Culture Machine* 13

Strebel, I., A. Bovet, and P. Sormani. 2018. *Repair Work Ethnographies: Revisiting Breakdown, Relocating Materiality*. London: Springer

Suchman, L., R. Trigg, and J. Blomberg. 2002. "Working Artefacts: Ethnomethods of the Prototype." *The British Journal of Sociology* 53, 2: 163-179

Thrift, N. 2006. "Re-Inventing Invention: New Tendencies in Capitalist Commodification." *Economy and Society* 35, 2: 279-306

Van Der Veen, V., Y. Fratantonio, M. Lindorfer, D. Gruss, C. Maurice, G. Vigna, H. Bos, K. Razavi, and C. Giuffrida. 2016. "Drammer: Deterministic Rowhammer Attacks on Mobile Platforms." *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*,1675-1689.

Xiao, Y., X. Zhang, Y. Zhang, and R. Teodorescu. 2016. "One Bit Flips, One Cloud Flops: Cross-VM Row Hammer Attacks and Privilege Escalation." *{USENIX} Security Symposium* 25: 19-35

1. Beck, U. 1992. *Risk Society: Towards a New Modernity*. London: SAGE; Perrow, C. 1984. *Normal Accidents: Living With High Risk Technologies*. Princeton: Princeton University Press ↩

2. Graham, S., and N. Thrift. 2007 "Out of Order: Understanding Repair and Maintenance." *Theory, Culture & Society* 24, 3: 1-25; Jackson, S. J. 2014. "Rethinking Repair." T. Gillespie, P. J. Boczkowski, K. Foot (eds.) *Media Technologies: Essays on Communication, Materiality, and Society* 221-239; Strebel, I., A. Bovet, and P. Sormani. 2018. *Repair Work Ethnographies: Revisiting Breakdown, Relocating Materiality*. London: Springer ↩
3. Coleman, G. 2012. *Coding Freedom: The Ethics and Aesthetics of Hacking*. Princeton: Princeton University Press; Kelty, C. 2008 *Two Bits: The Cultural Significance of Free Software*. Durham: Duke University Press; Coleman, G., and C. Kelty. 2017. "Hacks, Leaks, and Breaches." *LIMN* 8 ↩
4. For instance, see recently Joque, J. 2018. *Deconstruction Machines: Writing in the age of cyberwar*. Minneapolis: University of Minnesota Press; Dwyer, A. C. 2019. *Malware Ecologies: A Politics of Cybersecurity*. Doctoral dissertation: University of Oxford. ↩
5. A broader conception has however been the focus of recent research in Science and Technology Studies, notably in, Hommels, A., Mesman, J. and Bijker, W.E. 2014. *Vulnerability in Technological Cultures: New Directions in Research and Governance*. Cambridge: MIT Press ↩
6. Amoore's research has highlighted the significance of modalities in the realm of human security; Amoore, L. 2013. *The Politics of Possibility: Risk and Security Beyond Probability*, Durham: Duke University Press. On the other hand, the constraint of possible behaviours of technical systems has been a preoccupation of computer security since the 1970s. See, for example, Mackenzie, D., and G. Pottinger. 1997. "Mathematics, Technology, and Trust: Formal Verification, Computer Security, and the US Military." *IEEE Annals of the History of Computing* 19, 3: 41-59. ↩
7. I am taking 'normative' here in the very general sense suggested by Rouse 'a covering term for any phenomena for which it makes good sense to understand… as open to assessment, whether in terms of success and failure, correctness and incorrectness, appropriateness and inappropriateness, justification or lack thereof, right or wrong, justice or injustice, and so forth', Rouse, J. 2016 "Normativity". In J. Kiverstein (Ed.) *The Routledge Handbook of Philosophy of the Social Mind*. London: Taylor & Francis. 545-561. 546. ↩
8. Rheinberger, H.-J. 1997. *Toward A History of Epistemic Things: Synthesizing Proteins in The Test Tube*. Stanford: Stanford University Press ↩
9. See, for example, Van Der Veen, V., Y. Fratantonio, M. Lindorfer, D. Gruss, C. Maurice, G. Vigna, H. Bos, K. Razavi, and C. Giuffrida. 2016. "Drammer: Deterministic Rowhammer Attacks on Mobile Platforms." *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications*

Security,1675-1689.; Frigo, P., E. Vannacci, H. Hassan, V. van der Veen, O. Mutlu, C. Giuffrida, H. Bos, and K. Razavi. 2020. "TRRespass: Exploiting the Many Sides of Target Row Refresh." *IEEE Symposium on Security and Privacy* 747-762 ↩

10. Kim, Y., R. Daly, J. Kim, C. Fallin, J. Hye Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. 2014. "Flipping Bits In Memory Without Accessing Them: An Experimental Study Of DRAM Disturbance Errors." *ACM SIGARCH Computer Architecture News* 42, 3: 361-372. ↩
11. See, for example, Van Der Veen et al. "Drammer"; Xiao, Y., X. Zhang, Y. Zhang, and R. Teodorescu. 2016. "One Bit Flips, One Cloud Flops: Cross-VM Row Hammer Attacks and Privilege Escalation." *{USENIX} Security Symposium* 25: 19-35 ↩
12. See, for example, Bains, K., J. Halbert, C. Mozak, T. Schoenborn, and Z. Greenfield. "Row Hammer Refresh Command." U.S. Patent 9,117,544, issued August 25, 2015. ↩
13. Greenfield, Z., K. S. Bains, T. Z. Schoenborn, C. P. Mozak, and J. B. Halbert. "Row hammer Condition Monitoring." U.S. Patent 8,938,573, issued January 20, 2015. Not paginated. ↩
14. Kim et al. "Flipping Bits." ↩
15. JESD79-3F DDR3 SDRAM Standard (2012) ↩
16. Kim et al., "Flipping Bits," 361 ↩
17. Ibid ↩
18. Ibid ↩
19. Kim et al., "Flipping Bits," 363 ↩
20. Blanchette, J.-F. 2011. "A material history of bits." *Journal of the American Society for Information Science and Technology* 62, 6:1042-1057. 1047. We are of course here visiting themes raised by Kittler. 'To minimize all the noise that it would be possible to eliminate (i.e. via the containment and control of disturbance) is the prize (sic) paid for structurally programmable machines.' Kittler, F. 1995. "There Is No Software." *CTheory* 10-18.. 16. ↩
21. One might conceptualise these hybridities as 'epiphylogenetic memory' or 'cognitive assemblages'. See Stiegler, B. 2012. "Relational Ecology and the Digital Pharmakon." *Culture Machine* 13; Hayles, N. K. 2016. "Cognitive Assemblages: Technical Agency and Human Interactions." *Critical Inquiry* 43, 1: 32-55. ↩
22. Schaefer, R. 2009. "The Epistemology of Computer Security." *ACM SIGSOFT Software Engineering Notes* 34, 6: 8-10. ↩
23. Floridi, L., N. Fresco, and G. Primiero. 2015. "On Malfunctioning Software." *Synthese* 192, 4: 1199-1220. ↩

24. Seaborn, M., and T. Dullien. 2015. "Exploiting the DRAM Rowhammer Bug to Gain Kernel Privileges." *Black Hat* 15 ↩
25. Aga, M. T., Z. B. Aweke, and T. Austin. 2017. "When Good Protections Go Bad: Exploiting anti-DoS Measures to Accelerate Rowhammer Attacks." *IEEE International Symposium on Hardware Oriented Security and Trust (HOST)* , 8-13 ↩
26. Razavi, K., B. Gras, E. Bosman, B. Preneel, C. Giuffrida, and H. Bos. 2016. "Flip Feng Shui: Hammering a Needle in the Software Stack." *USENIX Security Symposium* 25: 1-18 ↩
27. Ibid ↩
28. Van der Veen et al. "Drammer." ↩
29. Knowlton, K. C. 1965. "A Fast Storage Allocator." *Communications of the ACM* 8, 10: 623-624. ↩
30. A recent review is given in Mutlu, O. and J. S. Kim. 2019. "RowHammer: A Retrospective." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* ↩
31. Frigo et al. "TRRespass" ↩
32. Ibid. ↩
33. Kwong, A., D. Genkin, D. Gruss, and Y. Yarom. 2020. "Rambleed: Reading bits in memory without accessing them." *IEEE Symposium on Security and Privacy* 41 ↩
34. Govindavajhala, S. and A. W. Appel. 2003. "Using Memory Errors to Attack a Virtual Machine." *IEEE Symposium on Security and Privacy*, 154-165 ↩
35. Mutlu & Kim "Rowhammer, a Retrospective." See Kocher, P., J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg et al. 2019. "Spectre Attacks: Exploiting Speculative Execution." *IEEE Symposium on Security and Privacy* 40; and Lipp, M., M. Schwarz, D. Gruss, T. Prescher, W. Haas, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg. 2018. "Meltdown: Reading Kernel Memory from User Space." *27th USENIX Security Symposium*. 18: 973-990. ↩
36. Robertson, J., and M. Riley. 2018. "The Big Hack: How China Used a Tiny Chip to Infiltrate US Companies." *Bloomberg Businessweek* 4 ↩
37. For instance, Elkins M. 2019. "Nation-State Supply Chain Attacks for Dummies and You Too." *CS3STHLM* 21-22 Oct. 2019. ↩
38. Bachelard, Gaston. 1968. *The Philosophy of No: A Philosophy of the New Scientific Mind*. Viking Press ↩
39. Bachelard, G. 1984. *The New Scientific Spirit*. Trans. A. Goldhammer. Boston: Beacon Press, 172 ↩
40. Hacking, I. 2004. *Historical Ontology*, Cambridge: Harvard University Press ↩

41. Suchman, L., R. Trigg, and J. Blomberg. 2002. "Working Artefacts: Ethnomethods of the Prototype." *The British Journal of Sociology* 53, 2: 163-179; Thrift, N. 2006. "Re-Inventing Invention: New Tendencies in Capitalist Commodification." *Economy and Society* 35, 2: 279-306; Neff, G., and D. Stark. 2004. "Permanently Beta." In Howard P. N. and Jones S. (eds.) *Society Online: The Internet in Context*, London: SAGE. 173-188; Chun, W. H. K. 2016. *Updating to Remain the Same: Habitual New Media*, Cambridge: MIT Press ↩
42. Chun, W. H. K. 2011. *Programmed Visions: Software and Memory*, Cambridge: MIT Press; Galloway, A. R. 2006 "Language Wants to be Overlooked: On Software and Ideology." *Journal of Visual Culture*, 5, 3: 315-331 ↩
43. Bogost, I. 2015. "The Cathedral of Computation", *The Atlantic*. Available at: https://www.theatlantic.com/technology/archive/2015/01/the-cathedral-of-computation/384300/ ↩
44. Jackson, "Rethinking." 223. ↩
45. Jackson, S. J. 2017. "Speed, Time, Infrastructure." J. Wajcmann, N. Dodd (eds.) *The Sociology of Speed: Digital, Organizational, and Social Temporalities* 169-186, 173. ↩
46. Star, S. L., and K. Ruhleder. 1996. "Steps Toward an Ecology of Infrastructure: Design and Access for Large Information Spaces." *Information Systems Research* 7, 1: 111-134. Frabetti has made a similar point in her deconstructive take on software studies (the credit goes to an anonymous reviewer for pointing me to this particular excerpt) 'For a computer professional, [writes Frabetti, ↩
47. Kauffman, S. 2000. *Investigations*. Oxford: Oxford University Press. 131 ↩
48. Felin, T., S. Kauffman, R. Koppl, and G. Longo. 2014. "Economic Opportunity and Evolution: Beyond Landscapes and Bounded Rationality." Strategic Entrepreneurship Journal 8, 4: 269-282 276 ↩
49. Houkes, W., & Vermaas, P. E. 2010. *Technical Functions: On the Use and Design of Artefacts* London: Springer ↩
50. Dennett, D. C. 1990. "The Interpretation of Texts, People and other Artifacts." *Philosophy and Phenomenological Research*, 50: 177-194 ↩